Claims

What is claimed is:

Claim 1:

1. An apparatus which employs the inventive enhancements to lottery scheduling of computer process tasks substantially as described herein.

Claim 2:

2. A method which employs the inventive enhancements to lottery scheduling of computer process tasks substantially as described herein.

Claim 3:

3. An operating system which employs the inventive enhancements to lottery scheduling of computer process tasks substantially as described herein.

Claim 4:

4. A scheduler which employs the inventive enhancements to lottery scheduling of computer process tasks substantially as described herein.

Claim 5:

5. A second stage lottery program for a dispatcher program within an operating system of a computer system wherein said second stage lottery system determines which of at least two tasks of at least two levels within a selected class of at least two task type classes will be assigned to a next available IP resource available to a scheduler queue on which pointers to said at least two tasks reside, said second stage lottery program comprising:

random number generator and selection program for generating a random number and for selecting a one of said at least two levels within said selected class based upon a correspondence of a thereby generated random number and

transfer program for transferring control from said second stage lottery program to a task found on said selected one of said at least two levels.

Claim 6:

6. The second stage lottery program of claim 5 further comprising:

a level switching routine for handling a failure by said transfer program to find a task on a selected one of said at least two levels, said level switching routine for modifying said level selecting of a one of said at least two levels by said random number generator and selection program to a second one of said at least two levels, iteratively, until a task is found on a selected level, and when a task is found, allowing transfer of control to said found task.

Claim 7:

7. The second stage lottery program of claim 5 wherein said at least two levels selected among by said random number generator and selection program are constructed wherein each next higher level among said at least two levels is two times more likely to be selected than its next lower level among said at least two levels.

Claim 8:

8. The second stage lottery program of claim 7 wherein each level of said at least two levels will only have tasks of like quantum values within said each level.

Claim 9:

9. The second stage lottery program of claim 5 wherein each of said tasks has a quantum value, said quantum value identifying a computer system specific amount of time in which said each of said tasks with said quantum value may continuously execute on an instruction processor resource, and wherein said second stage lottery employs a quantum bias routine, said quantum bias routine comprising:

a data capture routine for determining how much of the allotted segment of a quantum a task that has executed used before returning control to said dispatcher, and

a bias adjustment routine for adjusting a current allotted segment of a quantum identifying how much of said quantum said task that has executed has to a new allotted segment of a quantum for said a task.

Claim 10:

10. The second stage lottery program of claim 9 wherein said bias adjustment routine does not adjust said current allotted segment to a new allotted segment for said task that has executed if said task that has executed's use of its current allotted segment was interrupted by an interrupt.

Claim 11:

11. The second stage lottery program of claim 9 wherein said bias adjustment routing does not adjust said current allotted segment to a new allotted segment for said task that has executed if a use of a current allotted segment of said task that has executed was interrupted by an interrupt.

Claim 12:

12. A computer system having a quantum timer settable to allow processing on an IP resource for a limited duration by any task, also having an operating system having a dispatcher program wherein all said any tasks are identifiable as being members of classes and wherein said dispatcher program comprises:

a) A scheduler code section executable to determine to which of said any tasks pending in a scheduler queue that an IP resource will be next assigned to process on said IP resource and for how long said next assigned task may process on said IP resource, and

b) A scheduler queue from which said any tasks may be addressable and assignable to said IP resource, wherein

said scheduler code section has a two stage lottery execution algorithm, a first of said two stages for determining from which class of said classes a next one of said any tasks will be selected by a second stage lottery execution algorithm to process on said IP resource, said second of said two stages determining which among said any tasks that happen to be within said class will be said next assigned task.

Claim 13:

13. The computer system set forth in claim 12 wherein said first stage lottery execution algorithm performs to choose a class randomly but with a bias settable by a user among all said classes.

Claim 14:

14. The computer system set forth in claim 12 wherein a number determining how many there are of said classes is selectable by a user.

Claim 15:

15. The computer system as set forth in claim 12, wherein if any one of said any tasks is of a class "above the lottery line" then said any one of said any tasks is assigned an IP resource prior to running said scheduler code.

Claim 16:

16. The computer system as set forth in claim 12, wherein if the scheduler code section's first of two stages selects a class which is empty of said any tasks, said scheduler code section next chooses another class of available classes.

Claim 17:

17. The computer system as set forth in claim 12, wherein if none of the classes has any of said any tasks, said scheduler code selects a very low priority level operating system task.

Claim 18:

18. The computer system as set forth in claim 12, wherein said second stage lottery comprises:

random number generator and selection program for generating a random number and for selecting a one of said at least two levels within said selected class based upon a correspondence of a thereby generated random number, and

transfer program for transferring control from said second stage lottery program to a task found on said selected one of said at least two levels.

Claim 19:

19. A method for use by a dispatcher algorithm in an operating system in a computer system for selecting a task to provide with an available instruction processor resource wherein said method comprises:

determining whether within a scheduler queue there are any task pointers within priority levels of a class of tasks on said scheduler queue, and if so, determining whether any said any task pointers are above a second stage lottery line, and if so assigning a first of said any tasks indicated by said any task pointers above said second stage lottery line to said available instruction resource, but if not,

and if there is only one said priority level having any of said any task pointers, assigning a first of said any tasks corresponding to said any task pointers at said only one said priority level to said available instruction processor resource, else,

running a second stage lottery algorithm to determine to which priority level among a plurality of said priority levels having said any task pointers said available instruction processor resource should be made available to assign to a task within said which priority level.

Claim 20:

20. A method for use by a dispatcher algorithm in an operating system in a computer system for selecting a task to provide with an available instruction processor resource wherein said method comprises:

determining whether within a scheduler queue there are any task pointers within priority levels of a class of tasks on said scheduler queue, and if so, and if there is only one said priority level having any of said any task pointers, assigning a first of said any tasks corresponding to said any task pointers at said only one said priority level to said available instruction processor resource, else,

running a second stage lottery algorithm to determine to which priority level among a plurality of said priority levels having said any task pointers said available instruction processor resource should be made available to assign to a task within said which priority level.

Claim 21:

21. The method of claim 20 further comprising:

moving said task pointers within priority levels of a class of tasks on said scheduler queue, wherein each priority level can maintain a chain of said task pointers and wherein there are more than one of said priority levels, said task pointer moving process comprising:

maintaining a task assigned quantum which identifies for each task on said scheduler queue a set amount of instruction processor resource to which said each task is entitled upon being assigned to an instruction processor resource,

placing said each task into a said priority level based upon a value in said task assigned quantum for said each task, and

changing said each task priority level based on how much of said value in said task assigned quantum for said each task said each task used a last time said each task was assigned to an instruction processor resource.

Claim 22:

22. The method of claim 21 wherein an algorithm for determining whether said each task priority level should be changed based on said last time said each task was assigned to an instruction processor resource is: if said last time did not complete within said task assigned quantum, lower this task by one priority level and raise said value of said task assigned quantum by factor of 2, if said last time used less than said value of said task assigned quantum but more than a small portion of said task assigned quantum, leave level and portion the same, and if said last time used less than said small portion of said value of said task assigned quantum, increase this task's priority level by one and halve this task's task assigned quantum value.

Claim 23:

23. The method of claim 21 wherein said task assigned quantum portion is assigned for said each task prior to a said task pointer being on said scheduler queue.

Claim 24:

24. The method of claim 21 wherein a first stage lottery algorithm selects which of said classes of tasks on said scheduler queue will have tasks assignable to said available instruction processor resource.

Claim 25:

25. The method of claim 21 wherein a first stage lottery algorithm selects which of said classes of tasks on said scheduler queue will have tasks assignable to said available instruction processor resource, but only if there are no tasks in classes above a first lottery line, and if there are tasks in said classes above said first lottery line, providing said available instruction processor resource to said tasks in classes above said first lottery line in a priority order.

Claim 26:

26. A dispatcher system for use within a computer system, said computer system having an available instruction processor resource and a scheduler for storing pointers to tasks to be assigned quantums of said instruction processor resource, said dispatcher system comprising a two stage lottery system through which each task is assigned to an available instruction processor resource for a portion of a quantum assigned to said task.